Remarks

The above Amendments and these Remarks are in reply to the Office Action mailed March 9, 2006. A Petition for Extension of Time is submitted herewith, together with the appropriate fee. No fee is due for the addition of new claims.

Applicant acknowledges with thanks Examiner Burgess' assistance in granting an interview on August 8, 2006, during the course of which interview various features of the claimed embodiments were discussed, the substance of which is included fully herein.

I. Summary of Examiner's Rejections

Prior to the Office Action mailed March 9, 2006, Claims 1-15, 17, 20-21, 24-27 and 29-33 were pending in the Application. In the Office Action, all of the claims were rejected under 35 U.S.C. 102(b) as being anticipated by Zimowski et al. (U.S. Patent No. 6,944,655, hereafter Zim).

II. Summary of Applicant's Amendment

The present Response cancels Claims 20-21; amends Claims 1-12, 24-26 and 29; and adds new Claims 34-37, leaving for the Examiner's present consideration Claims 1-15, 17, 24-27 and 29-37. Reconsideration of the Application, as amended, is respectfully requested. Applicant respectfully reserves the right to prosecute any originally presented or canceled claims in a continuing or future application.

III. Claim Rejections under 35 U.S.C. §102(b)

In the Office Action mailed March 9, 2006, Claims 1-15, 17, 20-21, 24-27 and 29-33 were rejected under 35 U.S.C. 102(b) as being anticipated by Zim (U.S. Patent No. 6,944,655).

Claim 1

Claim 1 has been amended to more clearly define the embodiment therein. As amended, Claim 1 defines:

> *1. (Currently Amended) A system for session-based retrieval at a client system of content from a server system, comprising:*

a communication protocol that enables an asynchronous session-based connection between a client system and a server system, and allows the client system to send, within a session between the client system and the server system, a plurality of consecutively input query strings, to query the server system for string-based content;

a client object, in communication with a client software at the client system and with the communication protocol, wherein the client object receives additional characters from the client software, and as each character is being received, transmits to a server object at the server system a plurality of consecutive queries, within the same session, to retrieve content from the server system, wherein each consecutive query one of lengthens or shortens the query string by the additional characters, and forms an increasingly focused query string for retrieving matching content from the server system; and

a server object, in communication with the server system, and with the client object via the communication protocol, wherein the server object in response to receiving each consecutive query as it lengthens or shortens the query string by the additional characters automatically matches the increasingly focused query string against the content of the server system, and asynchronously returns increasingly matching content information to the client object for immediate use by the client system.

Claim 1, as currently amended, defines a system for session-based retrieval at a client system of content from a server system, comprising a communication protocol that enables an asynchronous session-based connection between the client system and the server system. The communication protocol allows the client system to query the server system for *string-based content*. A client object, in communication with a client software at the client system receives *additional characters* from the client software. As *each additional character is being received*, the client object transmits to a server object at the server system a plurality of consecutive queries, within the same session, to retrieve content from the server system, wherein each consecutive query one of *lengthens or shortens the query string by the additional characters*, and forms an increasingly focused query string. In response to receiving each consecutive query as it lengthens or shortens the query string by the additional characters, the server object automatically *matches the increasingly focused query string against the content of the server* system, and asynchronously returns increasingly matching content information to the client object for immediate use by the client system. Applicant respectfully submits that these features are not disclosed by the cited references.

The advantages of the embodiment currently defined by Claim 1 include that the server maintains a session-based connection with the client, so that the server knows immediately when a search string character is entered or deleted by an end-user at the client, without the end-user having to, e.g. click "submit" each time. The server can then respond automatically as each character is being received at the client object. Since the server can also asynchronously return increasingly matching content information to the client object for immediate use by the client system, whenever the server receives each new character from the client (again, as the end-user is entering characters into the client but without the end-user having to, e.g. click "submit"), it can match the newly modified search string with any potential matches within its content, and can communicate this information to the client as immediate feedback. This immediate feedback may be in the form of e.g. providing preliminary search results, or suggesting a more appropriate, or a better-matching content to the end-user. From the perspective of the end-user, the feedback resembles a form of auto-complete or auto-suggest, presented to the user as the user is actually typing the search string. This is a very user-friendly means of searching complex databases and data sources, but which to date has not been provided in a Web-environment.

Zim discloses a system and method for allowing a client application that is remotely connected to a server to invoke a stored procedure on the server and to have multiple query result sets returned to the client application on a single network message exchange. (Column 2, lines 1-7). In one embodiment, the system includes a client process that constructs a client process execution request identifying a stored procedure and specifying constraints on the quantity of response data that the client process is capable of handling. The client process execution request is transferred to a server process. The server process invokes the stored procedure identified by the client process execution request, and generates answer set data for a plurality of diverse query result sets. The server process then generates an initial response containing, for each of the query result sets, an amount of the obtained answer set data consistent with the constraints. The initial response is transferred from the server to the client process. The client application is then able to access and process answer set data for the diverse result sets in any order. (Column 2, line 59 - Column 3, lines 14). Zim further describes with respect to Figure 3, that the client process execution request includes information that: (1) specifies the name of the stored procedure to be executed; (2) indicates that the client process is prepared to receive one or more query result sets;

and (3) specifies certain constraints on the quantity of response data that the client process is capable of handling. The initial response includes a summary component and a plurality of query result set components. The client process receives the initial response and processes the summary component, and the query result set components. Subsequently, the client process processes requests from the client application to obtain access to the answer set data for the query result sets. In doing so, the client process may interact with the server process to obtain additional answer set data for one or more of the query result sets. (Column 5, line 40 - Column 8, lines 67).

Applicant respectfully submits that, as described above, Zim discloses a client-server environment in which a plurality of server database objects and result sets are exchanged with a client using a minimal number of network messages. The advantages of such a system are (i) it involves a *single exchange* of messages between the client and the server; (ii) both the client and the server can *constrain* the amount of data that will be sent to the client; and (iii) once some initial amount of data is received, the client can subsequently select, in any order, which *additional data* it may want to retrieve.

However, Applicant respectfully submits that this technique differs substantially from that defined by Claim 1, as currently amended. As defined by Claim 1, a client can repeatedly send *multiple queries* to the server to retrieve data from the server, rather than a single exchange of messages. Since this approach allows for multiple queries or requests, such an approach would be undesirable in Zim which aims to reduce the overall number of messages.

Furthermore, as defined by Claim 1, when the server receives queries it *asynchronously* returns increasingly matching content information to the client object for immediate use by the client system. Unlike Zim, this feature allows data to be returned to the user as the user is actually entering a search string.

Furthermore, unlike Zim, the relevance of data, and the size of the data returned to each client can be determined by the server, in large part by the total number of 'hits' that match the client query and the preferred size of data sets that the client would like to handle. In some of the present embodiments the server may decide not to send back any data for one of the plurality of queries received from the client, instead opting to only send back replies for the final query and thus avoiding the unnecessary traffic and load that would be caused by the earlier query in the session.

To consider these points in greater detail, Applicant respectfully submits that, as described above, Zim discloses a client-server environment in which *a plurality of server database objects* is exchanged with a client using a minimal number of network messages. By contrast, Claim 1 defines an embodiment in which a client can repeatedly retrieve *an ever-more-focused content* from a server based on a previously defined query, wherein the content is matched/mapped directly to the input string query as it is being formed.

Unlike Zim, which discloses database objects in a SQL client-server environment, embodiments of the present invention use *strings of characters* themselves (which grow in size as the user types the characters in the string), and more particularly consecutive queries that one of lengthen or shorten the query string by the additional characters, and that form an increasingly focused query string for retrieving matching content from the server system.

As further described above, in Zim, a number of *different stored procedures to be executed at the server are identified by each consecutive query* from the client. One or more stored procedures are executed once, and the client can then selectively retrieve partial or full result sets for the combined query. Typically, the server returns an initial response containing the results for these one or more stored procedures. Thus, each consecutive query identifies a *different* stored procedure. Furthermore, Zim discloses that if the client process needs *additional data* from *any of the result sets*, then the client process can send a message to the server process specifying the amount of additional query result set data to be returned. This suggests that the prior request from a client has asked for one or more different result sets, and that the second and subsequent requests from the client communicate *an indicator to return part of a previous result set's data*. Zim does not appear to disclose a system whereby multiple consecutive string-based queries from a client system are applied against the same content and the same native queries for the underlying content source.

However, as defined by Claim 1, the client object receives additional characters from the client software, and *as each character is being received*, transmits to a server object at the server system a plurality of consecutive queries, within the same session, *to retrieve content from the server system,* and that each consecutive query one of lengthens or shortens the query string by one or more *additional characters*. Thus, as defined by Claim 1, the content source is typically already identified by the client system during an initial client-server message within the session., and

that same content source is then used to match the additional characters against. Unlike Zim, the content source and the resulting corresponding native queries performed on the server do not usually change at any time within the session for a specific client object. Consecutive queries only contain the growing string of characters (rather than a set of stored procedures with corresponding parameters. As such, in the embodiment defined by Claim 1, the input string has a direct relationship to the content being matched on the server - once a client has registered with the server by starting a session (at which time the name of the Content Source to be used is specified to the Server), then subsequent requests from that client simply pass along the user data (the "growing string of characters"), rather than names of additional/different stored procedure(s) with parameters to be executed.

Furthermore, as also defined by Claim 1, each consecutive client request or query one of lengthens or shortens the query string by the additional characters, and forms an increasingly focused query string for retrieving matching content from the server system. Thus, each query contains a different query string that *lengthens or shortens the previous one*, rather than, as disclose in Zim, an indicator to return part of a previous result set's data. Indeed, as described in Zim, it does not seem possible to modify the system therein to extend a prior query in order to get "narrowed down" results, and the consecutive queries cannot be lengthened or shortened by one or more characters in order to create a focused query string that is matched against the content of the server system.

In view of the above comments, Applicant respectfully submits that Claim 1, as currently amended, is neither anticipated by nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claims 20, 21, 24 and 29

The comments provided above with respect to Claim 1 are hereby incorporated by reference. Claims 20 and 21 have been canceled, rendering moot the rejection of these claims. Claims 24 and 29 have been similarly amended to more clearly define the embodiments therein. For similar reasons as provided above with respect to Claim 1, Applicant respectfully submits that Claims 24 and 29, as amended, are likewise neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claims 2-15, 17, 25-27 and 30-31

Applicant respectfully submits that the remaining Claims 2-15, 17, 25-27 and 30-31 should be allowable as depending upon an allowable independent claim, and further in view of the comments provided above. In addition, several of Claims 2-15, 17, 25-27 and 30-31 have been amended to more clearly define the embodiments therein, as further described below.

In particular, Claim 2 defines an embodiment in which the client object operates on or at a first computer and said server object operates on or at a second computer, and wherein both of said first and the second computers are connected via the communication protocol.

Claim 3 defines an embodiment in which the server object and the client object both run on the same computer. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Applicant respectfully submits that Zim discloses that the client computer system and the server computer system *are each implemented using a computer system*, and further illustrates in Figure 1 that these two computer systems are separated by a communication medium. Thus, Zim does not appear to disclose any embodiment in which the server object and the client object both run on the *same computer*.

Claim 4 defines an embodiment in which the system comprises *a plurality of server objects* that run on a plurality of separate computers, and wherein the client queries are distributed over the separate computers. Zim does not appear to disclose such an arrangement.

Claim 5 defines an embodiment in which the server object stores previously received results from the server as stored results, and initially returns the stored results to the client in response to new client queries, without accessing the content at the server. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Applicant respectfully submits that Zim discloses that the server therein may store a plurality of *stored procedures*, which are different from the *stored strings* defined by Claim 5.

Claim 6 defines an embodiment in which the client software is embedded into a software application that provides a visual interface to an operator and *allows the operator to add or remove additional characters to lengthen or shorten the query string*, while simultaneously receiving increasingly matching results from the server. Zim does not appear to disclose such an arrangement.

Claim 7 defines an embodiment in which the client software is used as a content engine for another software system.

Claim 8 defines an embodiment in which the client software accumulates a plurality of the single character queries as they are entered into the client, before sending the plurality of the single character queries together to the server as a single string. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim does not appear to disclose a system whereby multiple consecutively input characters are grouped into *one string-based query*. Instead, Zim describes that the server-side stored procedure may include a plurality of *database query instructions*.

Claim 9 defines an embodiment in which the client object *stores previously received responses from the server in a cache* and uses these as the response to a new query by the user, without re-accessing the server. Zim does not appear to disclose such an arrangement.

Claim 10 defines an embodiment in which the client software stores a pre-defined query string and automatically transmits it to the server as the client software is first accessed, and wherein additional entry of query characters is not required before server responses are sent to the client. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim does not appear to specify an *initial query* (*the "pre-defined string"*) to be sent and performed by the server when the client software is first accessed. Instead, Zim discloses only that the client receives the initial results as they are returned for any query.

Claim 11 defines an embodiment in which the server stores the state of query and response of the client software, and restores the state of the client software after any interruption in the communication protocol, including an automatic or manual network interruption or termination of the session. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim describes how the client system, after performing a query and receiving results, can ask for additional parts for the "answer set data" by specifying "the result set generator identifier for the query result set of interest". This suggests that in Zim, the client knows (and remembers) the result set identifier, and simply asks for additional data while the client/ server session is active. Zim does not appear to disclose that the server restores *the entire state of the client* (including both query and response). Nor does Zim disclose that server-side results and "result set identifiers" are cached or reused in separate (or restored) sessions.

Claim 12 defines an embodiment in which the client software adds a qualifier to the string query that is passed to the server, whereby the server can use the qualifier to execute the query and return appropriate results based on both the query string and its qualifier. In the office action, it was submitted that Zim discloses such an arrangement. However, Zim refers to stored procedures identified on the client by (1) name, accompanied by (2) an indication "that the client process is prepared to receive one or more query result sets", and (3) specifying "certain constraints on the quantity of the response data...". Thus, Zim describes named stored procedures, rather than strings (consisting of consecutively input characters) with optional qualifiers. Nowhere does Zim refer to an additional (or any) qualifier being sent to the server by the client.

Claim 13 defines an embodiment in which the client software identifies a user of the system to the server whereby the server can store statistics and provides a history of queries and corresponding responses appropriate to the user. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim discloses a system in which a client can request "query result sets" for its current query, whereby the server process identifies the "semantics" of each result set to the client so that the client can apply appropriate logic when handling these result sets. Zim does not appear to disclose a system that keeps a "history of queries and corresponding responses appropriate to the user", nor does Zim specify that the server can "store statistics".

Claim 14 defines an embodiment in which the server system comprises a server tier and a syndication tier, and wherein the client software communicates to the server tier on a single computer, and wherein each query is forwarded by the server tier and the syndication tier to an appropriate syndicate of content channels connected to the server tier on a different computer. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim discloses how the "initial response" and "supplemental response" are requested by the client system and transmitted from the server system. This is different from Claim 14, which describes that a first server may access content channels on other servers, whereby data is "syndicated" from one server to another. In this configuration, a first server becomes a "content engine" for another a second server.

Claim 15 defines an embodiment in which the server applies a content dependent pattern and filter to characters received from the client before queries are matched against the content.

In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim describes how the server creates an "initial response", and what it contains. By contrast, Claim 15 defines how the QuestObjects system can apply a pattern and filter to the string-based queries received from the client in order to transform the user query before it is matched against the server-side content engine(s). This would only work for string-based queries whose semantics remain the same during the user session, and cannot be applied to systems such as those described by Zim, which work with stored procedures rather than string-based queries.

Claim 17 defines an embodiment in which server responses comprise lists of strings, wherein each string is accompanied by corresponding metadata, whereby the metadata contains logical links to other data sources or Uniform Resource Identifiers (URLs). In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim describes how multiple "query result sets", corresponding to the parts of the stored procedure and each having different semantics, are returned to the client as part of the "summary component". By contrast, Claim 17 defines an embodiment in which results consist of lists of strings, whereby each string is accompanied by corresponding metadata, and whereby the metadata contains URLs. In Zim, the client may send various queries calling various stored procedures, each stored procedure returning various "types" of result sets, whereby the client must "decode" those results before processing and showing them to the user. In contrast, the present system normalizes all data to strings. Therefore, unlike Zim the client system does not need to associate the query result set with the logic (code) in the client application.

Claim 25 defines an embodiment wherein several input fields in the user interface have session connection indicators and status indicators to indicate to the user the availability of a connection between said client application and said content server for those input fields, and the status of increasingly available content at said content server for selection by said user at those input fields. In the Office Action, it was submitted that Zim discloses such an arrangement. Claim 25 has been amended herein, which may render the rejection moot, however, the term "input field" appears to be beyond the scope of the Zim patent.

Claim 26 defines an embodiment in which the session connection indicator displays a triangular display element to indicate the presence of the connection, and does not display the triangular display element to indicate the absence of the connection. In the Office Action, it was

submitted that Zim discloses such an arrangement. However, Zim only refers to information kept in memory in the client system; not the way in which one can visually indicate the status of the session to the user.

Claim 27 defines an embodiment in which the status indicator displays one, or a plurality of, arrow display elements to indicate the transfer of data from the client application to the server during the session, and the presence of available session-specific content at the server.

Claim 30 defines an embodiment in which the server object matches each query received from the client against an in-memory cache, and returns cached content to the client without accessing the content engine, unless the cached content has expired since it was last received from the content engine. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim discloses a system for fetching from the server selected elements from the result set for the current query. In contrast, Claim 30 defines an embodiment which caches the result sets for multiple queries received from any number of clients, and then returns appropriate cached results to subsequent identical requests coming in from any clients thereafter. This is a very different caching mechanism from that described in Zim.

Claim 31 defines an embodiment in which the server analyzes the time between the consecutive queries received from each client system, and skips selected ones of the consecutive queries to reduce network communications and the load on the content engine. In the Office Action, it was submitted that Zim discloses such an arrangement. However, Zim discloses a system where partial result sets are returned to the client rather than full result sets. In contrast, Claim 31defines a system where the server does not return any data to the client at all, because the client query has been or potentially will be superseded by a consecutive query from the same client.

Applicant respectfully submits that Zim does not appear to disclose or suggest the particular combination of features defined by each of the claims above. As such Applicant respectfully submits that Claims 11-14, 16-20, 22, 27 and 31-34 are neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claims 32 and 33

Applicant respectfully traverses the rejection of Claims 32 and 33. In the Office Action, Claims 32 and 33 were rejected under 35 U.S.C. 102(b) as being anticipated by Zim. However, several of the features of Claims 32 and 33, namely a user interface at the client that allows a user to enter a search string; a client object that receives characters of the search string from the user interface as it is being entered by the user, and transmits them to a server object at the server; a server object that automatically matches the search string against the content of the server system, and asynchronously returns increasingly relevant content information to the client object for immediate use by the client; and *wherein the content information is used by the client to immediately update the user interface with options that match the content of the server system, as the user is entering the search string*, were not explicitly addressed in the Office Action. As described above, Applicant respectfully submits that Zim fails to disclose these features. In view of the above comments, reconsideration of Claims 32 and 33 is respectfully requested.

IV. Additional Amendments

Claims 34-37 have been newly added by the present Response. Applicant respectfully requests that new Claims 34-37 be included in the Application and considered therewith.

V. Conclusion

In view of the above amendments and remarks, it is respectfully submitted that all of the Claims now pending in the subject patent application should be allowable, and reconsideration thereof is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if he can assist in any way in expediting issuance of a patent.

Enclosed is a PETITION FOR EXTENSION OF TIME UNDER 37 C.F.R. §1.136 for extending the time to respond up to and including September 11, 2006.

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: September 11, 2006

By: _____
Karl Kenna
Reg. No. 45,445

Customer No.: 23910
FLIESLER MEYER LLP
Four Embarcadero Center, Fourth Floor
San Francisco, California 94111-4156
Telephone: (415) 362-3800